UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/824,450	04/14/2004	Adam H. Leventhal	03226.417001;SUN040872	6601

32615 7590 06/13/2008

OSHA LIANG L.L.P./SUN
1221 MCKINNEY, SUITE 2800
HOUSTON, TX 77010

EXAMINER	
LEE, MARINA	

ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
06/13/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lord@oshaliang.com
DOCKETING@OSHALIANG.COM
hathaway@oshaliang.com

Office Action Summary	Application No.	Applicant(s)	
	10/824,450	LEVENTHAL ET AL.	
	Examiner	Art Unit	
	MARINA LEE	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *15 February 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,7-11 and 13* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4,7-11 and 13* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *02/15/2008*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to Amendments dated February 15, 2008.

Claims 1-3 and 8 have been amended.

Claims 5, 6, and 12 have been cancelled.

Accordingly, claims 1-4, 7-11, and 13 are presented for examination.

2. Examiner withdraws the Double Patenting Rejection in view of Applicants' filling Terminal disclaims to comply with 37 CFR § 1.321c.

3. Applicants' arguments for the claims have been fully considered but they are not persuasive, as will be also addressed under Prior Art's Arguments – rejections section at item (5) below. Thus, the rejection of the claims over prior art in the previous Office Action is maintained in light of the necessitated additional clarification provides hereon to the amended claims and accordingly, **THIS ACTION IS MADE FINAL**. See MPEP §706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Prior Art's Arguments – Rejections

4. Applicant's arguments filed on February 15, 2008 have been fully considered but they are not persuasive.

As to claim 1, which now amended to include the feature of claims 5 and 6, Applicants contend that "the Examiner admits that Tamches fails to specifically discloses searching a look-up table using a program counter value...In view of this, Tamches fails to discloses all the limitation " See Remarks, page 6, ¶ 1, which examiner strongly agrees; however, it is to note that, the limitation "searching a look-up table using a program counter value" was previously cited in claim 5, which was rejected under 103 rejection by Tamches in view of Mattson, in the previous Office action (page 6).

Furthermore, Applicants allege that "claim 1 requires, in part, loading the original instruction into a scratch space, which is allocated on per-thread basis. In contrast, Tamches only discloses that a single code patch and a single springboard, if necessary, are allocated when splicing an instruction. .. In other words, all thread branched by the instruction will be directed to the same code path (or to the same springboard). Further, Tamches discloses a "safer approach" where a counter reflects the number of thread currently in the same code patch... Thus, Tamches discloses that multiple threads may exist in the same code patch simultaneously. However, there is now disclosure of a scratch space allocated on a per thread basis. In view of this, it is clear that splicing allocating a single code patch of Tamches is not equivalent to the scratch space

allocated on a per-thread basis as recited in amended independent claim 1" – *See Remarks, page 6, ¶ 2 with emphasis added*, is not persuasive.

First of all, the limitation of the amended claim 1, that Applicants allege for recites "loading the original instruction into a scratch space, wherein the scratch space is allocated on a per-thread basis". As can be seen, the Applicants' allegation "Tamches discloses a "safer approach" where a counter reflects the number of threads currently in the same code patch" as cited above, is irrelevant to the limitation of the recitation claim since there is no mention of "counter" in the claim.

Secondly, the claim merely states "loading the original instruction into a scratch space, wherein the scratch space is allocated on a per-thread basis" does not explicitly prevent one from interpreting the claims as "allocated on a per-thread needed" meaning "the spring board" of Tamches contains number of thread(s) and those thread (s) to be allocated as needed.

Therefore, Tamches in view of Mattson does disclose the amended claim 1 recitation.

As to the amended independent claim 8, Examiner notes that Applicants call for similar argument as to claim 1 state above – See Remarks, page 7, ¶ 3 and page 8, ¶ 2, but fail to be found persuasive. Further, Applicants argue that "Mattson is completely silent with respect to a look-up table comprising information related to the original instruction. In view of this, Mattson does not teach or suggest a look-up table comprising an original instructions associated with a probe and address associated with

the original instructions " – See Remarks, page 7, last paragraph, which Examiner strongly disagrees.

Mattson discloses "for purposes of illustration, it will be assumed that driver 310 fetches an original instruction, referred to hereafter as the current original instruction, from original instruction storage area 201. Driver 310 then passes control to look-up table comparator 315. Using any of a variety of known techniques, such as search and compare techniques, comparator 315 compares the unique identifier of the current original instruction to a list of unique identifiers in translated instruction look-up table 232. As is described below, table 232 includes unique identifiers that identify original instructions that have been translated and placed in translated instruction storage area 202. Multi-branch backpatcher 1020 determines, using any of a variety of known techniques, such as search and compare techniques, whether such current original multi-branch jump target address has been entered by memory manager 720 into translated instruction look-up table 232. If there is such an entry, it will be assumed for illustrative purposes that the address in translated instruction area 202 corresponding to the original address represented by "address2" is represented by the label TR_ADD2. As noted, the existence of such an entry indicates that such address was part of a hot trace that previously has been translated and optimized and stored in translated instruction storage area 202" *see at least col. 12: 1-12 and col. 28: 65-67 and col.29: 1-8, with emphasis added.*

As of the forgoing discussion above, Tamches in view of Mattson does disclose the limitations of claims 1 and 8.

As to the remaining claims directly or indirectly depends upon the independent claims1 and 8 – (*See Remarks page 6, ¶3, and page 8, last paragraph*) are also fall together as Applicants relied upon rebuttal for the independent claims but fail to be found persuasive as noted above.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. C**laims 1-4, and7-11, and 13** are rejected under 35 U.S.C. 103(a) as being unpatentable over Tamches, "Fine-Grained Dynamic Instrumentation of Commodity Operating System Kernels – of record", University of Wisconsin, 2001 and in view of Mattson, Jr. et al., (hereinafter – Mattson), (U.S. Patent No. 6,327,704 B1 of record).

As to claim 1, Tamches discloses, a method of tracing an instrumented program on a processor having an x86 architecture, comprising:

triggering a probe in the instrumented program (*see –e.g., "probe point" of IBM's Dynamic probe code (Dprobes), page 15, "instrumentation point" of code patch, page 49, and "entry point" of the original code to unconditional branch, page 62*);

obtaining an original instruction associated with the probe (see "instrumentation point" of code patch ,page 49, and "entry point" of the original code to unconditional branch, page 62) ;

loading the original instruction into a scratch space (*e.g., allocate the code patch and data heaps are load into kernel address space using driver /dev/kernist; and springboard (scratch space) – see KernInst Architecture 3.1, Page 26-27 and Fig. 4.9, and page 62*), wherein the scratch space is allocated on a per-thread basis – see pages 51, 62 and related text).

loading a jump instruction for the x86 architecture into the scratch space wherein the jump instruction includes a next program counter value (*e.g., overwrite a single branch or jump instruction x86 by writing a one-byte trap or .. .and jump to the appropriate code patch – see page 66-67, section 4.6.1 and code splicing overwrite instruction of Figure 4.9, page 62*);

executing the original instruction in the scratch space using a thread (*e.g., the code patch is executed at boot time or run-time of the kernel and the springboard execution – page 49, page 62, and associated text*); and

executing the jump instruction in the scratch space using the thread (*e.g., the code patch is executed at boot time or run-time of the kernel and the springboard execution – page 49, page 62, and associated text*).

It is noted that Tamches does not explicitly disclose wherein obtaining the original instruction comprises: searching a look-up table using a program counter value, wherein the look-up table comprises the original instructions associated with the probe and an address associated with the original instruction. However, Mattson, in an analogous art, teaches dynamic translator using branches of the multi-branch-jump instruction are executed, the dynamic backpatching code enables a backpatcher that

replaces the corresponding entry in the translated multiple branch-jump table using with pointers to the address of the translated target address, e.g., direct jump backpatch 1010 determines, using any of a variety of know techniques, such as search and compare techniques, whether such ordinal target address has been enter by memory manager 720 in translated instruction look-up table 232 (see Mattson, Abstract, col. 12: 1-51, Look-up table 232, col. 27: 31-42, and related text).

It would have been obvious to a person having ordinary skill in the art at the time of invention was made to use dynamic backpatching determination through look-up table 232 of Mattson in dynamic instrumentation (e.g., handle to look up address) of Tamches for increasing the efficiencies of the dynamic translation as once accomplished by Mattson (see Mattson, col. 1: 48-63).

As to claim 2, Tamches discloses further comprising:

emulating the original instruction to determine the program counter value if the original instruction is a control-flow instruction; and returning control to the thread at an address of the program counter value if the original returning control to the thread at an address of the program counter value if the original instruction is control-flow instruction (e.g., the relocating the overwritten instruction to the code patch semantic section 4.2.1, page 51-52).

As to claim 3, Tamches discloses further comprising:

determining the next program counter value by incrementing the program counter value using a size of the original instruction (e.g., increment the program counter for

each original function call or event counter increment – *see Figure 5.1, page 72 and associated text*).

As to claim 4, Tamches discloses wherein the probe corresponds to a trap (e.g., trap handle use in splicing and trap instruction – see page 15-17 and section 4.6.1, page 66-67).

As to claim 7, wherein the instrumented program is executed on a multi-thread architecture (e.g., even interval accumulation per multiple thread – see pages 51, 62 & 66 and related text).

As to claim 8, Tamches discloses a system for tracing an instrumented program on a processor having an x86 architecture, comprising:

a thread configured to execute the instrumented program (*e.g., allocate the code patch and data heaps are load into kernel address space using driver /dev/kernist; and springboard (scratch space) – see KernInst Architecture 3.1, Page 26,-27 and Fig. 4.9, and page 62*);

a scratch space arranged to store the original instruction and the jump instruction (*e.g., kernel address space and springboard (available space) – see KernInst Architecture 3.1, Page 26,-27 and Fig. 4.9, and page 62*) , wherein the scratch space is allocated on a per-thread basis – see pages 51, 62 and related text).

; and

an execution facility for executing the original instruction in the scratch space to collect data and executing the jump instruction, wherein the execution facility is a processor based on the x86 architecture (*e.g., overwrite a single branch or jump*

instruction x86 by writing a one-byte trap or .. .and jump to the appropriate code patch – see page 66, section 4.6.1 and code splicing overwrite instruction of Figure 4.9, page 62).

It is noted that Tamches does not explicitly disclose a look-up table arranged to store an address and a corresponding original instructions; a trap handler configured to halt execution of the thread when a trap instruction is encountered, use an address of the trap instruction to obtain the corresponding original instruction from the look-up table, and generated a jump instruction to an address in the instrumented program trap. . However, Mattson, in an analogous art, teaches dynamic translator using branches of the multi-branch-jump instruction are executed, the dynamic backpatching code enables a backpatcher that replaces the corresponding entry in the translated multiple branch-jump table using with pointers to the address of the translated target address, e.g., direct jump backpatch 1010 determines, using any of a variety of know techniques, such as search and compare techniques, whether such ordinal target address has been enter by memory manager 720 in translated instruction look-up table 232 (see Mattson, Abstract, col. 12: 1-51, Look-up table 232, col. 27: 31-42, and related text).

It would have been obvious to a person having ordinary skill in the art at the time of invention was made to use dynamic backpatching determination through look-up table 232 of Mattson in dynamic instrumentation (e.g., handle to look up address) of Tamches for increasing the efficiencies of the dynamic translation as once accomplished by Mattson (see Mattson, col. 1: 48-63).

As to claim 9, Tamches discloses further comprising: a buffer for storing the data (e.g., L1 data cache see page 84 or kernel space Figure 3.1, or springboard space, Figure 4.9, page 62).

As to claim 10, Tamches discloses further comprising:

a tracing framework configured to emulate the original instruction to determine a value of a program counter if the original instruction is a control-flow instruction and to return control to a thread at an address of the program counter value if the original instruction is a control-flow instruction (e.g., the relocating the overwritten instruction to the code patch semantic section 4.2.1, page 51-52).

As to claim 11, Tamches further discloses wherein the trap handler sets a destination of the jump instruction to a next address immediately following and address of the trap instruction (*e.g., overwrite a single branch or jump instruction x86 by writing a one-byte trap or…. and jump to the appropriate code patch – see page 66-67, section 4.6.1 and code splicing overwrite instruction of Figure 4.9, page 62).*

As to claim 13, Tamches further discloses wherein the instrumented program is executed on a multi-thread architecture (e.g., even interval accumulation per multiple threads – see page 66 and related text).

Conclusion

7. The prior art made of record and not relied upon is considered pertinent to the applicant disclosure.

Hsu et al. (US 6,295,644 B1) is cited to teach patching program text to improve performance of application.

Pierce et al., "Idtrace – A Tracing Tool for i486 Simulation", University of Michigan, IEEE, 1994, is cited to each Idtrace, a binary instrumentation tool which produces execution traces for the ix86 instruction set architecture.

Bosch et al., "Complete x86 instruction trace generation from hardware bus collect", University of Paris (emphasis added), IEEE, 1997, is cited to teach hardware/software approach to collect perfect x86 traces using a commercial analyzer.

Uhlig, "Trace-Driven Memory Simulation: A survey", University of Michigan, ACM, 1997, is cited to teach surveys and analyzes the trace-driven memory simulation by establishing criteria for evaluating, and then applies the criteria those criteria".

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Marina Lee whose telephone number is (571) 270-1648. The examiner can normally be reached on M-F (11:00 am to 7: 30 pm) EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/M. L. /
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192